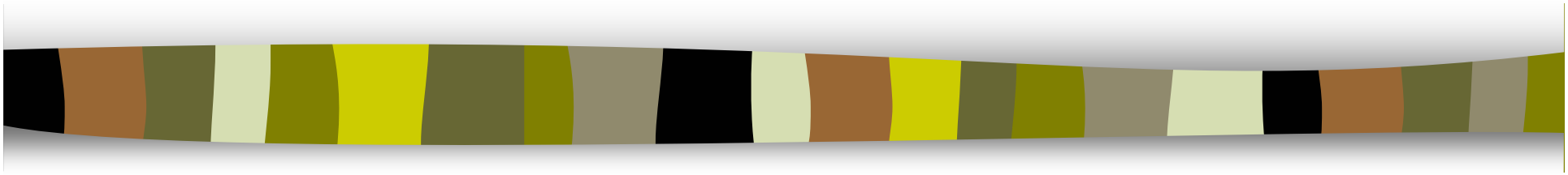
Curso de PHP



Tema 2: Lenguaje PHP básico

José Mariano González Romano
mariano@lsi.us.es

SOME RIGHTS RESERVED

Tema 2: Lenguaje PHP básico

1. Sintaxis básica
2. Tipos de datos
3. Variables
4. Constantes
5. Expresiones y operadores
6. Estructuras de control
7. Funciones
8. Tablas
9. Bibliotecas de funciones

Sintaxis básica

- PHP es sensible a las mayúsculas
- ¿Cómo se incrusta en la página web?

  `<?PHP ... ?>`

  recomendado, siempre disponible

  `<?= expresión ?>`

  equivale a <? echo expresión ?>
- Las instrucciones se separan con un ; como en C. La marca final ?> implica un ;
- Comentarios: como en C, /* … */ y //

Sintaxis básica

- Para imprimir: **echo** y **print**

  echo: muestra una o más cadenas
  echo cadena1 [, cadena2…]; // no es una función

```
echo “Hola mundo”;
echo “Hola “, “mundo”;
```

  print: muestra una cadena
  print cadena; // no es una función

```
print “Hola mundo”;
print  “Hola “ . “mundo”;
```

Sintaxis básica

- Ejemplo:

```
<HTML>
<HEAD>
<TITLE>Mi primer programa en PHP</TITLE>
</HEAD>

<BODY>

<?PHP
   print (“<P>Hola mundo</P>”);
?>

</BODY>
</HTML>
```

Sintaxis básica

- Uso de \n para generar código HTML legible
- a) Sin \n

Código PHP

```
print (“<P>Párrafo 1</P>”);
print (“<P>Párrafo 2</P>”);
```

Código HTML

```
<P>Párrafo 1</P><P>Párrafo 2</P>
```

Salida

```
Párrafo 1

Párrafo 2
```

Sintaxis básica

- Uso de \n para generar código HTML legible
- b) Con \n

Código PHP

```
print (“<P>Párrafo 1</P>\n”);
print (“<P>Párrafo 2</P>\n”);
```

Código HTML

```
<P>Párrafo 1</P>
<P>Párrafo 2</P>
```

Salida

```
Párrafo 1

Párrafo 2
```

Sintaxis básica

- **Ejercicio 1: programa que muestra un mensaje**
  - Ilustra cómo incrustar código PHP en un documento HTML y cómo imprimir desde PHP

Sintaxis básica

- Inclusión de ficheros externos:
  - **include**()
  - **require**()
- Ambos incluyen y evalúan el fichero especificado
- Diferencia: en caso de error include() produce un warning y require() un error fatal
- Se usará require() si al producirse un error debe interrumpirse la carga de la página
- Ejemplo:

Sintaxis básica

```
<HTML>
<HEAD>
   <TITLE>Título</TITLE>
<?PHP
// Incluir bibliotecas de funciones
   require ("conecta.php");
   require ("fecha.php");
   require ("cadena.php");
   require ("globals.php");
?>
</HEAD>
<BODY>
<?PHP
   include ("cabecera.html");
?>
// Código HTML + PHP
. . .
<?PHP
   include ("pie.html");
?>
</BODY>
</HTML>
```

Tipos de datos

- PHP soporta 8 **tipos de datos primitivos**:
  - Tipos escalares: boolean, integer, double, string
  - Tipos compuestos: array, object
  - Tipos especiales: resource, NULL
- El tipo de una variable no se suele especificar. Se decide en tiempo de ejecución en función del contexto y puede variar
- Funciones de interés:
  - La función gettype() devuelve el tipo de una variable
  - Las funciones is_*type* comprueban si una variable es de un tipo dado:

    is_array(), is_bool(), is_float(), is_integer(), is_null(), is_numeric(), is_object(), is_resource(), is_scalar(), is_string()
  - La función var_dump() muestra el tipo y el valor de una variable. Es especialmente interesante con los arrays

Tipos de datos

- Tipo **integer** (números enteros)
  - 27, -5, 0
- Tipo **double** (números reales)
  - 1.234, -5.33
- Tipo **boolean** (lógico)
  - Valores: *true*, *false* (insensibles a las mayúsculas)
  - El 0 y la cadena vacía tienen valor *false*

Tipos de datos

- Tipo string:
  - Las cadenas se encierran entre comillas simples o dobles:
    - ‘simples’: admite los caracteres de escape \’ (comilla simple) y \\ (barra). Las variables **NO** se expanden
    - “dobles”: admite más caracteres de escape, como \n, \r, \t, \\, \$, \”. Los nombres de variables **SÍ** se expanden
    - Ejemplos:

```
$a = 9;
print ‘a vale $a\n’;
   // muestra a vale $a\n
print “a vale $a\n”;
   // muestra a vale 9 y avanza una línea
print “<IMG SRC=‘logo.gif’>”;
   // muestra <IMG SRC=‘logo.gif’>
print “<IMG SRC=\”logo.gif\”>”;
   // muestra <IMG SRC=“logo.gif”>
```

  - Acceso a un carácter de la cadena:
    - La forma es $inicial = $nombre{0};

Variables

- Las variables siempre van precedidas de un $
- El nombre es sensible a las mayúsculas
- Comienzan por letra o subrayado, seguido de letras, números o subrayado
- Variables predefinidas:

  $GLOBALS, $_SERVER, $_GET, $_POST, $_COOKIES, $_FILES, $_ENV, $_REQUEST, $_SESSION
- Ámbito: globales al fichero (excepto funciones) o locales a una función
- Ejemplo:

```
$valor = 5;
print “El valor es: “ . $valor . “\n”;
print “El valor es: $valor\n”; // ojo: comillas dobles

Resultado:
   El valor es: 5
```

Variables

- Variables variables
  - Se pueden crear nombres de variables dinámicamente
  - La variable variable toma su nombre del valor de otra variable previamente declarada
  - Ejemplo:

```
$a = "hola";
$$a = "mundo";

print "$a $hola\n";
print "$a ${$a}";

Resultado:
      hola mundo
      hola mundo
```

Variables

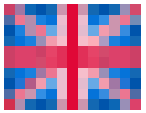
- Ejemplo de variables variables: página internacionalizada (1)

```
<?PHP
   $mensaje_es="Hola";
   $mensaje_en="Hello";
   $idioma = "es";
   $mensaje = "mensaje_" . $idioma;
   print $$mensaje;
?>
```

Variables

- Ejemplo de variables variables: página internacionalizada (2)

```
<?PHP
   $mensaje_es="Hola";
   $mensaje_en="Hello";
   $idioma = "en";
   $mensaje = "mensaje_" . $idioma;
   print $$mensaje;
?>
```



Constantes

- Definición de constantes:

```
define (“CONSTANTE”, “hola”);
print CONSTANTE;
```

- No llevan $ delante
- Sólo se pueden definir constantes de los tipos escalares (boolean, integer, double, string)

Expresiones y operadores

- Operadores aritméticos:
  +, -, *, /, %, ++, --
- Operador de asignación:
  =
  operadores combinados: .=, +=, etc
  $a = 3; $a += 5; → a vale 8
  $b = “hola ”; $b .= “mundo”; → b vale “hola mundo”
  → Equivale a $b = $b . “mundo”;
- Operadores de comparación:
  ==, !=, <, >, <=, >= y otros
- Operador de control de error: @. Antepuesto a una expresión, evita cualquier mensaje de error que pueda ser generado por la expresión
- Operadores lógicos:
  and (&&), or (||), !, xor
  and/&& y or/|| tienen diferentes prioridades
- Operadores de cadena:
  concatenación: . (punto)
  asignación con concatenación: .=

Expresiones y operadores

- Precedencia de operadores (de mayor a menor):

```
++, --
*, /, %
+,-
<, <=, >, >=
==, !=
&&
||
and
or
```

Estructuras de control

- Estructuras selectivas:
  - if-else
  - switch
- Estructuras repetitivas:
  - while
  - for
  - foreach

Estructuras de control

- Estructura selectiva **if-else**

```
if (condición)
   sentencia
```

```
if (condición)
   sentencia 1
else
   sentencia 2
```

```
if (condición1)
   sentencia 1
else if (condición2)
   sentencia 2
...
else if (condición n)
   sentencia n
else
   sentencia n+1
```

- Mismo comportamiento que en C
- Las sentencias compuestas se encierran entre llaves
- elseif puede ir todo junto

Estructuras de control

- Ejemplo de estructura selectiva if-else:

```
<?PHP
   if ($sexo == ‘M’)
         $saludo = "Bienvenida, ";
   else
         $saludo = "Bienvenido, ";
   $saludo = $saludo . $nombre;
   print ($saludo);
?>
```

Estructuras de control

- Estructura selectiva **switch**

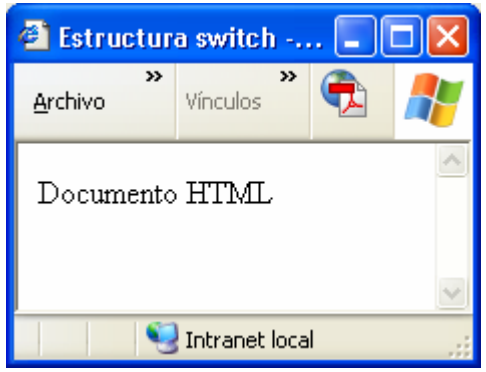
```
switch (expresión)
{
   case valor_1:
         sentencia 1
         break;
   case valor_2:
         sentencia 2
         break;
   …
   case valor_n:
         sentencia n
          break;
   default
         sentencia n+1
}
```

- Mismo comportamiento que en C, sólo que la expresión del case puede ser integer, float o string

Estructuras de control

- Ejemplo de estructura selectiva switch:

```
switch ($extension)
{
   case ("PDF"):
      $tipo = "Documento Adobe PDF";
      break;
   case ("TXT"):
      $tipo = "Documento de texto";
      break;
   case ("HTML"):
   case ("HTM"):
      $tipo = "Documento HTML";
      break;
   default:
      $tipo = "Archivo " . $extension;
}
print ($tipo);
```



Estructuras de control

- Estructura repetitiva **while**

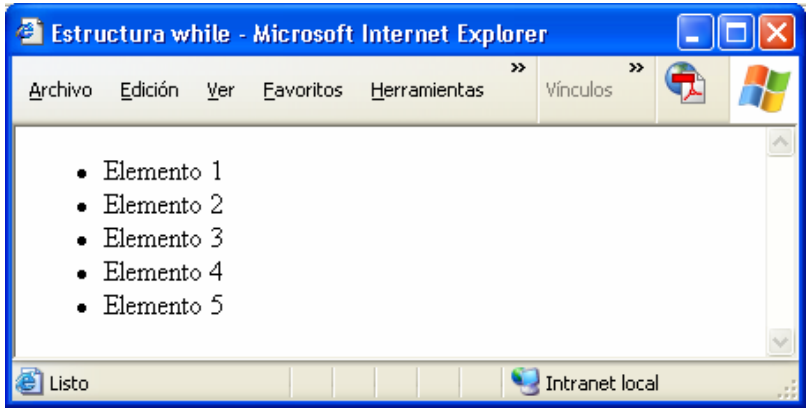
```
while (condición)
   sentencia
```

- Mismo comportamiento que en C

Estructuras de control

- Ejemplo de estructura repetitiva while:

```
<?PHP
   print ("<UL>\n");
   $i=1;
   while ($i <= 5)
   {
         print ("<LI>Elemento $i</LI>\n");
         $i++;
   }
   print ("</UL>\n");
?>
```



Estructuras de control

- Estructura repetitiva **for**

```
for (inicialización; condición; incremento)
   sentencia
```

- Mismo comportamiento que en C

Estructuras de control

- Ejemplo de estructura repetitiva for:

```
<?PHP
   print ("<UL>\n");
   for ($i=1; $i<=5; $i++)
         print ("<LI>Elemento $i</LI>\n");
   print ("</UL>\n");
?>
```

Estructuras de control

- **Ejercicio 2: programa que calcula una tabla de multiplicar**
  - Ilustra cómo manejar variables y cómo usar bucles

Funciones

- Ejemplo:

```
function suma ($x, $y)
{
   $s = $x + $y;
   return $s;
}

$a=1;
$b=2;
$c=suma ($a, $b);
print $c;
```

Funciones

- Por defecto los parámetros se pasan por valor
- Paso por referencia:

```
function incrementa (&$a)
{
   $a = $a + 1;
}

$a=1;
incrementa ($a);
print $a; // Muestra un 2
```

Funciones

- Argumentos por defecto

```
function muestranombre ($titulo = "Sr.")
{
    print "Estimado $titulo:\n";
}
muestranombre ();
muestranombre ("Prof.");
```

- Salida:

```
Estimado Sr.:
Estimado Prof.:
```

Funciones

- Los argumentos con valores por defecto deben ser siempre los últimos:

```
function muestranombre ($nombre, $titulo= "Sr.")
{
    print "Estimado $titulo $nombre:\n";
}
muestranombre (“Fernández”);
muestranombre (“Fernández”, "Prof.");
```

- Salida:

```
Estimado Sr. Fernández:
Estimado Prof. Fernández:
```

Tablas

- Sintaxis:
```
array ([clave =>] valor, ...)
```
- La clave es una cadena o un entero no negativo. El valor puede ser de cualquier tipo válido en PHP, incluyendo otro array
- Ejemplos:
```
$color = array (‘rojo’=>101, ‘verde’=>51, ‘azul’=>255);
$medidas = array (10, 25, 15);
```
- Acceso:
```
$color[‘rojo’] // No olvidar las comillas
$medidas[0]
```
- El primer elemento es el 0

Tablas

- La estructura de control **foreach** permite iterar sobre arrays
- Sintaxis:

```
foreach (expresión_array as $valor)
   sentencia
foreach (expresión_array as $clave => $valor)
   sentencia
```

- Ejemplos:

```
foreach ($color as $valor)
   print “Valor: $valor<BR>\n”;
foreach ($color as $clave => $valor)
   print “Clave: $clave; Valor: $valor<BR>\n”;
```

- Salida:

```
Valor: 101
Valor: 51
Valor: 255
Clave: rojo; Valor: 101
Clave: verde; Valor: 51
Clave: azul; Valor: 255
```

Bibliotecas de funciones

- Existen muchas bibliotecas de funciones en PHP
- Algunos ejemplos:
  - Funciones de manipulación de cadenas
  - Funciones de fecha y hora
  - Funciones de arrays
  - Funciones de ficheros
  - Funciones matemáticas
  - Funciones de bases de datos
  - Funciones de red
- Algunas bibliotecas requieren la instalación de componentes adicionales
- Todas las funciones de biblioteca están comentadas en la documentación de PHP

Bibliotecas de funciones

- Funciones de manipulación de cadenas
  - explode()
    - Divide una cadena en subcadenas
    - array **explode** (string separator, string string [, int limit])
  - rtrim(), ltrim(), trim()
    - Eliminan caracteres a la derecha, a la izquierda o por ambos lados de una cadena
    - string **rtrim** (string str [, string charlist])
  - strstr()
    - Busca la primera ocurrencia de una subcadena
  - strtolower() / strtoupper()
    - Convierte una cadena a minúscula / mayúscula
  - strcmp() / strcasecmp()
    - Compara dos cadenas con/sin distinción de mayúsculas
  - strlen()
    - Calcula la longitud de una cadena

Bibliotecas de funciones

- Funciones de fecha y hora
  - date()
    - Formatea una fecha según un formato dado
    - Ejemplo:

```
$fecha  = date ("j/n/Y H:i");
print ("$fecha");
Resultado:
        26/9/2005 17:36
```

  - strtotime()
    - Convierte una fecha en un *timestamp* de UNIX
    - Ejemplo:

```
$fecha = date ("j/n/Y", strtotime(“5 april 2001"));
print ("$fecha");
Resultado:
        5/4/2001
```

Bibliotecas de funciones

- Funciones de arrays
  - array_count_values()
    - Calcula la frecuencia de cada uno de los elementos de un array
  - array_search()
    - Busca un elemento en un array
  - count()
    - Cuenta los elementos de un array
  - sort(), rsort()
    - Ordena y reindexa un array (r=decreciente)
  - ksort(), krsort()
    - Ordena por claves un array (r=decreciente)

Bibliotecas de funciones

- **Ejercicio 3: programa que muestra la fecha actual**
  - Ilustra cómo usar comentarios, tablas y funciones (propias y de biblioteca). También cómo usar el manual de PHP